Week 2 - Part 1

Environments

Purpose

The purpose of this text is to introduce beginners to programming environments and their components, providing an understanding of their significance and influence on code development and execution. It covers essential elements like the operating system, programming language, runtime environment, virtual machine, development tools, as well as popular tools like Visual Studio Code, Node.js, and npm, and the online platform Replit.

What is an environment?

When working with programming you will come across the word “environment” a lot.
It can be difficult to explain and to understand, because it is a concept that can be applied to many different areas within programming.
It can also be specific applications or software.
As an example, if you are on a Mac computer, you work in a MacOS environment.
But, if you work in an integrated development environment, or IDE, that is also an environment.

In the general context of programming, an environment refers to the combination of hardware, software, and settings in which a program runs or is developed.
It encompasses everything necessary for the code to execute properly and produce the desired results. Understanding the programming environment is essential because it influences how code is written, tested, and executed. Different environments may have different requirements, capabilities, and performance characteristics, so it's crucial to choose the appropriate environment for the specific programming task at hand.
A programming environment typically includes the following components that we will discuss!

OS - Operating System

The underlying software that manages computer hardware and provides services for other software programs.
It handles tasks such as memory management, process scheduling and file system operations.

Programming language

Each programming language has its own syntax, rules, and libraries, which determine how the code is written and executed. For most languages you need to download the language, in order to provide the proper environment on your computer for the language to run.

Part 1 Environments

Programming languages are typically not directly executable by the computer's hardware. Instead, they require a compiler or interpreter to translate the code into machine-readable instructions. A compiler translates the entire program into machine code, while an interpreter executes the code line by line. You also download libraries and frameworks for that specific language. Libraries are prewritten code modules that provide additional functionality, such as data manipulation, networking, or user interface elements.
Frameworks are more extensive collections of libraries and tools that offer a structured way of building applications.

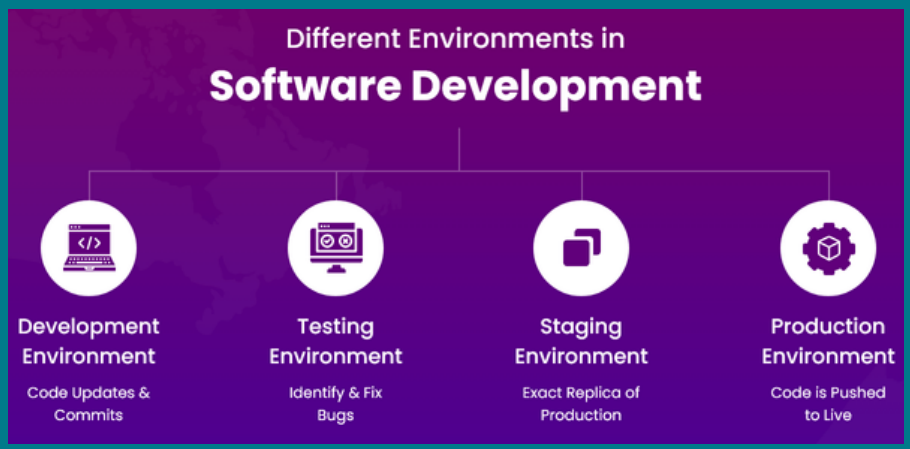
Runtime Environment

Runtime environments hide low-level details (more advanced features that are operate in the background in order for software to run properly) and provide a consistent execution environment for programs written in programming languages. This allows developers to focus on writing code rather than managing system-level operations.
They take care of the nitty-gritty, technical tasks that the program needs to run, but you as a coder don't need to deal with.

When we say runtime environments provide a, consistent execution environment, it means they give a reliable and predictable place for your code to run.
No matter where your program ends up, someone else's computer, or a server somewhere around the world, the runtime environment ensures that it will behave the same way.

Runtime environments are like the backstage crew in a theater production.
They make sure everything runs smoothly when the program is happening.
They manage all the tasks you don't see but are necessary for the program to work.

Each software environment serves a distinct purpose in the software development lifecycle, from initial coding and testing to final deployment for public use. The progression through these environments helps ensure that software is examined, leading to more reliable and high-quality products.



Part 1 Environments

Runtime environments are often specific to certain programming languages, often consisting of the necessary software and resources required to execute programs written in a particular language. It includes the runtime system, libraries, and dependencies specific to that language, providing the necessary infrastructure and support for running the code.
It abstracts the complexities of the underlying hardware and provides a standardized platform for developers to write and run their applications.
An example of this is Node.js, a popular open-source runtime environment for executing JavaScript code.

Virtual Machines (VMs)

A virtual machine is a software emulation of a physical computer system. It allows you to run multiple operating systems or environments within a single physical machine. Each virtual machine operates independently and has its own operating system as well as all the code needed for that virtual machine to run. This mean that the computer you re working on has an operating system, but the virtual machine has an additional operating system.

While it tends to take up more memory and repeat code, it's very useful for testing and isolating different environments.

Development Tools

These are tools that aid in the creation, editing, and debugging of code. This includes text editors, IDEs, debuggers, profilers, and version control systems.
We always work with the latest release, so we have an opportunity to learn, experiment and play with future features.

By utilizing virtual machines, users can quickly create and discard multiple isolated environments, minimizing the risk of software conflicts and facilitating safe experimentation with new configurations.

Part 1 Environments

When code is going to be used by the public it's best to use the stable version to ensure that it has been tested and is working and that most errors have been caught.
We are going to work with JavaScript, so we will be working with Node as our environment in the future.

However, for this course we are working with REPLIT, so you don't have to spend time on setting up the environment.
Instead, you will do that when you have time to dedicate to that task and when you have access to people that can give you more individualized help.
Right now, we can focus on starting coding and set you up for success when the course starts!

A development environment is a combination of hardware, software, and settings that allows programmers to write, test, and execute code.

The environment abstracts complexities and ensures code behaves consistently, making it easier for developers to focus on writing and running their applications. There are even online environments, like Replit, so you can code without setup!

IDEs

VS Code is the code editor we will be using for future courses. This also has a built-in terminal that can be used exactly as any other terminal. You can also choose any terminal to work in VS Code. That means that you can for example open Git Bash in VS Code . In that instance, it looks integrated into VS Code, but you are still actually working in Git Bash. This is important to note for the future when you might have many different terminals open.
An Integrated Development Environment, an IDE is a software application that provides a comprehensive set of tools and features to aid software development.

So what does that mean? Think of an IDE as a word processor for code, with a lot more features than just writing the code. It's a centralized platform that combines various tools, such as code editors, compilers, debuggers, and build automation, into a single user interface.
Here are some key features and functionalities of an IDE.

Part 1 Environments

Code Editor

IDEs provide a code editor with syntax highlighting that helps with writing a given programming language, auto-completion, cleanup code and provides code formatting features.
It offers a friendly environment for writing and editing code, making it easier to spot errors and maintain consistency.

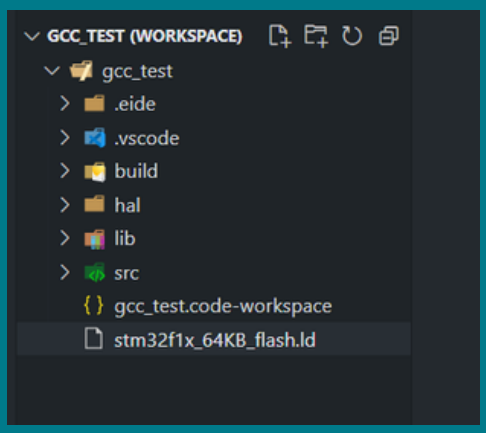
Compilation and Debugging.

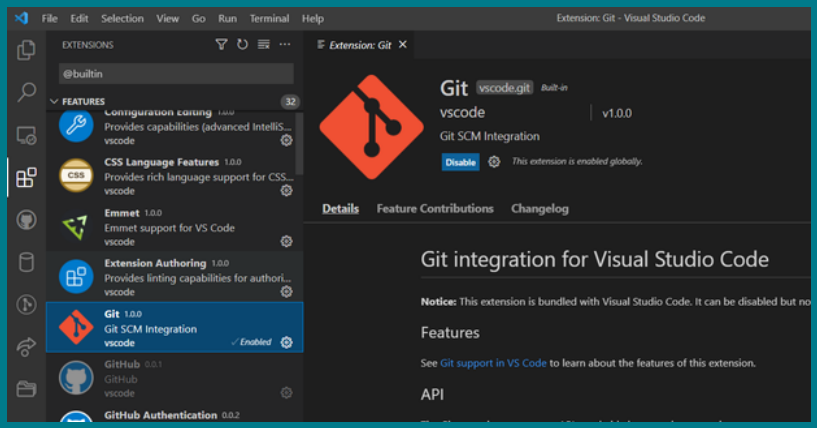
IDEs often include a compiler or interpreter to compile and run the code directly within the IDEs. They also provide debugging tools that help developers identify and fix issues in their code.
This debugging allows you to slow down the code, so that you can check what happens after each line of code.
When debugging, you mark where you want the IDEs to take a break and stop running the code, in order to examine exactly what happens.

Project Management

IDEs facilitate project organization by allowing developers to create and manage projects with multiple files, folders. They provide tools for creating new files and folders, and navigating between different parts of the project.

VSCode visually represents the real folder structure. You open the editor inside a folder. This way, VSCode provides visual way for developers to navigate and organize their files and folders inside the IDE.





Git can be installed as an extension, providing a visual UI for git operations! Extensions span from vanity extensions to very useful and necessary extension that makes a coders life much easier!

Part 1 Environments

Version Control Integration

Many IDEs integrate with version control systems like Git, allowing developers to manage repositories, view version history, and perform version control operations, such as commit, pull, and push, directly within the IDE.

Build and Deployment Tools

IDEs often include tools to automate building applications and assist with deployment. Deployment means the process of delivering the code onto a platform where users can use the software. IDEs can help compile code, package applications, and even deploy them to specific environments or platforms.

VS Code

At school we use the free IDE VSCode, made by Microsoft for Windows, MacOS and Linux.
However, there are many others, including some that are general and some that are specialized for a specific coding language.
Generally, you can choose to work in any IDE you want. However, the school examples are in VSCode, so it will be easier to follow along.
Also, VSCode is industry standard and is among the most popular out there!
Most tutorials on youtube use VSCode, and in general it is considered a staple in the programming community.

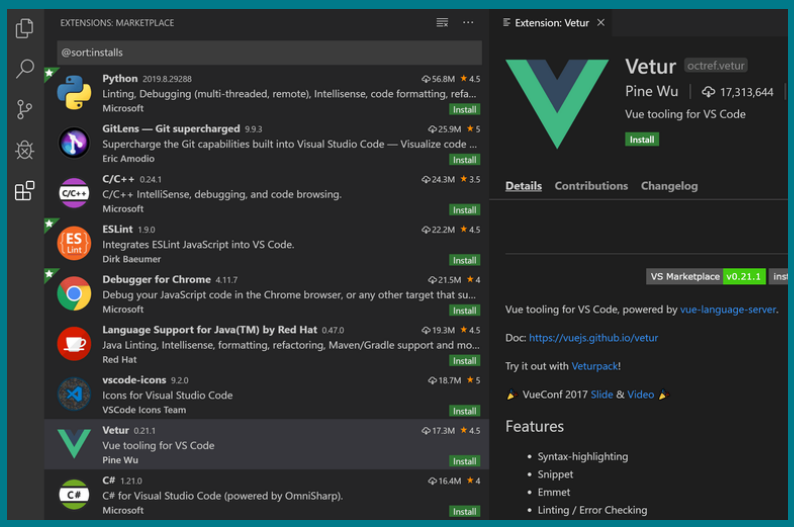
Extensions

VS Code as it is, comes as a blank slate, with only basic functionalities.
In order for everyone to be able to customize the user experience, one can download extensions.
These are small programs that only contain small features that integrate with VS Code.

It allows you to add new features, languages support, tools, themes, and more to customize and tailor your development environment according to your needs.
Extensions in VS Code are created by the community and third-party developers, expanding the capabilities of the editor beyond its out of the box features.
They are typically distributed through the Visual Studio Code Marketplace, where you can search, browse, and install extensions.
One example is Dracula, which changes the UI coloring of the program, or VS Code icons, that create icons for all your folders and files. One can also get extensions that work like spell check for your coding language, such as IntelliSense which is provided for JavaScript, TypeScript, JSON, HTML and CSS.

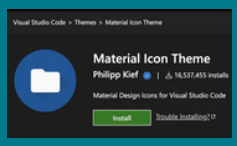
Part 1 Environments

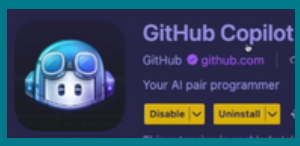
The reason these are not included straight away is so that for example Python users don't have extensions that only work with JavaScript.
By doing this VS Code manages to run faster and smoother, and users get to customize their experience!
If you find a good, useful or fun extension for VS Code, don't hesitate to share it in Slack!

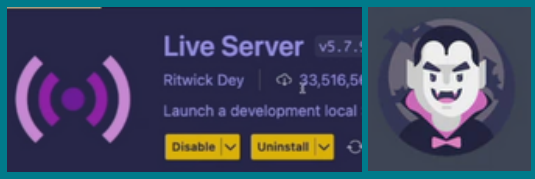


The extensions menu in VSCode, where you can search for extensions, install them and get some notes on how they work!

There are so many extensions out there, and it's pretty easy to find people recommending extensions for all programmer levels! Get your feet wet and do a quick you tube search for good extensions for beginner programmers!







Node.js

Node.js or Node is an open-source JavaScript runtime environment that allows developers to execute JavaScript code outside the web browser.
JavaScript has traditionally only been for the browser, but this has changed.
With node, developers can use JavaScript to build server applications, handle network requests, perform file operations, interact with databases, and more on their computer.
Overall, it is a runtime environment that creates a controlled environment for executing JavaScript programs, providing the necessary resources and services to run the code effectively.
It is built on the vee eight JavaScript engine, which is the same engine used by the Google Chrome web browser.

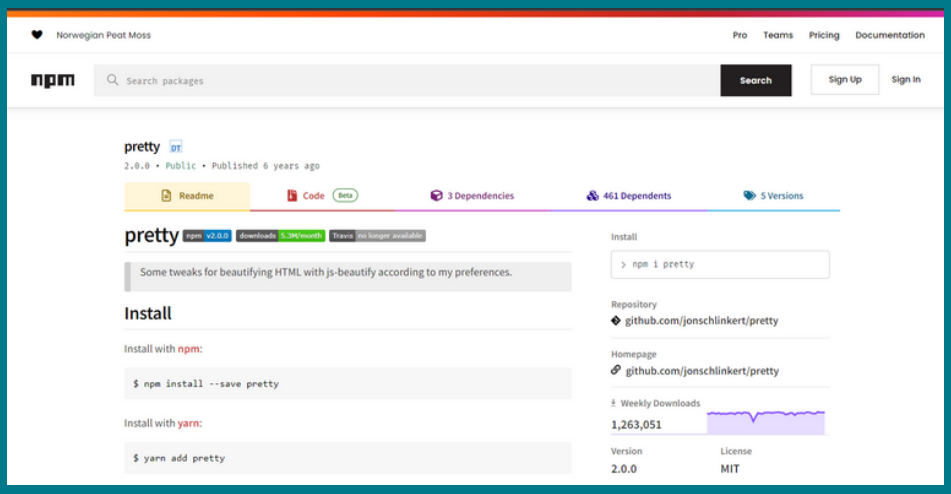
npm

Node.js has a rich ecosystem of packages and modules available through npm.
npm stands for Node Package Manager.
It is a package manager for the JavaScript programming language, specifically designed for use with the Node.js runtime environment.
With npm you can download JavaScript packages to help you program!

It acts as a software tool that automates various tasks related to installing, updating, configuring, and removing software packages on a computer system.
In the context of software development, a package is a bundled collection of code, assets, and metadata that can be easily distributed and installed.

These packages are designed to provide specific functionality or solve specific problems faced by developers. You can think of npm packages as similar to extensions.
They allow you to customize your JavaScript experience by adding functionality and solving problems without modifying the core features of Node.js.
npm provides a vast repository of packages that developers can explore, install, and utilize in their projects.

By leveraging npm packages, developers can enhance their development workflow, save time, and benefit from the expertise of the broader JavaScript community.
It enables smooth sailing by extending the capabilities of Node.js while keeping the core environment simple and efficient.



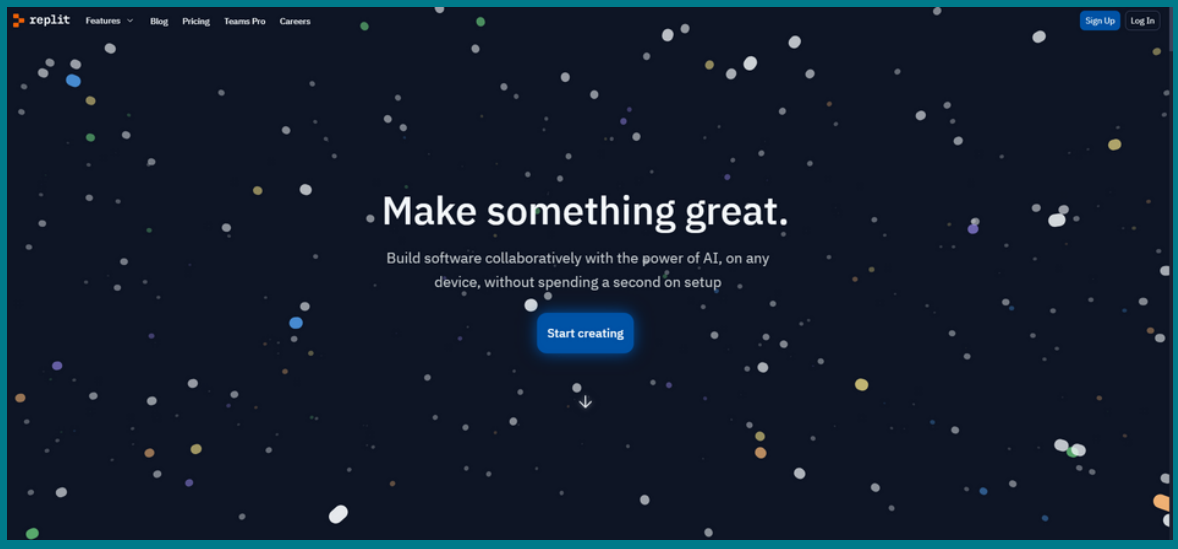
npm website, and how it looks when you click on a package. Make sure to check the weekly downloads! Remember, these are 3rd party packages, and a big community usually means its well loved!

REPLIT

We will be using REPLIT for this course.
REPLIT is an online coding and development platform that provides an integrated development environment in the cloud. It's basically VS Code online.
It offers a range of programming languages and tools, allowing users to write, run, and collaborate on code directly from their web browsers. REPLIT provides a seamless and accessible coding experience, particularly for beginners or those looking for a quick and hassle-free coding environment.

REPLIT offers a fully featured code editor with syntax, programming grammar, highlighting, autocompletion, and code formatting for various programming languages.
It supports a wide range of programming languages, including Python, JavaScript, Java, C++, Ruby, and more. We will be working with JavaScript!
REPLIT provides an integrated console where users can execute their code and see the output directly within the browser.
Overall, REPLIT simplifies the coding experience by providing a web-based IDE with collaborative features, making it accessible for individuals or teams to write, run, and share code seamlessly.

REPLIT



Summary of Part 1

This text provides a comprehensive introduction to the concept of "environments" in programming and their pivotal role in software development. It explains that an environment encompasses the combination of hardware, software, and settings in which a program runs or is developed. It also explores the significance of development tools like code editors, compilers, debuggers, and version control systems in facilitating efficient code creation and maintenance. The focus is on Visual Studio Code, a widely used integrated development environment (IDE) that streamlines the development process with its comprehensive set of tools and features. Additionally, the text introduces Node.js, an open-source JavaScript runtime environment that extends JavaScript's usage beyond web browsers, allowing developers to build server applications and perform various operations on their computers. The role of npm (Node Package Manager) is also discussed, an ecosystem of packages and modules for developers to enhance their JavaScript development experience. Finally, the text introduces Replit, an online coding and development platform that provides an integrated development environment in the cloud, enabling users to write, run, and collaborate on code directly from web browsers.